N95-17286

11087\ 33273

TDA Progress Report 42-119 November 15, 1994 P-11

A Simplified Integer Cosine Transform and Its Application in Image Compression

M. Costa and K. Tong
Communications Systems Research Section

A simplified version of the integer cosine transform (ICT) is described. For practical reasons, the transform is considered jointly with the quantization of its coefficients. It differs from conventional ICT algorithms in that the combined factors for normalization and quantization are approximated by powers of two. In conventional algorithms, the normalization/quantization stage typically requires as many integer divisions as the number of transform coefficients. By restricting the factors to powers of two, these divisions can be performed by variable shifts in the binary representation of the coefficients, with speed and cost advantages to the hardware implementation of the algorithm. The error introduced by the factor approximations is compensated for in the inverse ICT operation, executed with floating point precision. The simplified ICT algorithm has potential applications in image-compression systems with disparate cost and speed requirements in the encoder and decoder ends. For example, in deep space image telemetry, the image processors on board the spacecraft could take advantage of the simplified, faster encoding operation, which would be adjusted on the ground, with high-precision arithmetic. A dual application is found in compressed video broadcasting. Here, a fast, high-performance processor at the transmitter would precompensate for the factor approximations in the inverse ICT operation, to be performed in real time, at a large number of low-cost receivers.

I. Introduction

The integer cosine transform (ICT) [1,2] is an approximation of the discrete cosine transform. It can be implemented exclusively with integer arithmetic, with associated advantages in cost and speed for hardware implementations. One particular version of the ICT has been chosen for the compression algorithm of the image telemetry to be transmitted from Jupiter by the Galileo spacecraft [3].

As part of an image-compression system, the role of the ICT is to decorrelate the picture elements of image blocks, typically of size 8×8, for subsequent quantization and entropy encoding. In this article, a simplified version of the ICT is investigated. The simplification involves approximating integer divisions by right shifts in the binary representations of the numbers and allows for faster and simpler hardware realizations. This version of the ICT is an orthogonal transform and is only approximately normalized. The departure from exact normalization is of little consequence since it can be compensated for in the inverse ICT operation, performed with real arithmetic.

In the next section, we review the calculation of the ICT and its interface with the quantization stage in an image compression system. In Section III, we look at two examples of quantization arrays. Then, in Section IV, we discuss the application of the modified algorithm to the compression of a set of planetary images. Concluding remarks are given in Section V.

II. The ICT Operation in Image Coding

We initially consider the ICT of a one-dimensional data vector $\mathbf{X}$ of size 8. We choose the ICT version adopted for the Galileo image telemetry compression. This ICT is implemented by premultiplying $\mathbf{X}$ by the orthogonal matrix C, given by

$$C = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 5 & 3 & 2 & 1 & -1 & -2 & -3 & -5 \\ 3 & 1 & -1 & -3 & -3 & -1 & 1 & 3 \\ 3 & -1 & -5 & -2 & 2 & 5 & 1 & 3 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 2 & -5 & 1 & 3 & -3 & -1 & 5 & -2 \\ 1 & -3 & 3 & -1 & -1 & 3 & -3 & 1 \\ 1 & -2 & 3 & -5 & 5 & -3 & 2 & -1 \end{pmatrix} \tag{1}$$

One attractive characteristic of the ICT is that the absolute values of its coefficients are equal to powers of two, or powers of two plus one. Thus, the matrix multiplication can be efficiently executed by shifts and additions.

Denoting the ICT vector by $\mathbf{Y}$, we write $\mathbf{Y} = C\mathbf{X}$. As C is an orthogonal matrix, we have $CC^T = D$, where D is diagonal. Now, let Δ denote the inverse of D. Clearly, we can factor the identity matrix I as $I = \sqrt{\Delta}CC^T\sqrt{\Delta}$. Thus, we can identify $M = \sqrt{\Delta}C$ as an orthonormal matrix, so that $M^T = C^T\sqrt{\Delta} = M^{-1}$. The matrix M represents the normalized ICT.

In source-compression applications, the ICT coefficients of image blocks are typically individually quantized, and the entire block is then entropy encoded. The quantization can be implemented by rounded integer divisions of the transformed coefficients by quantization factors. The rounded integer division of a coefficient a by a factor q is given[1] by $\lfloor (a/q) + 0.5 \rfloor$, where $\lfloor x \rfloor$ denotes the largest integer smaller than or equal to x. Since normalization and quantization involve integer divisions, it makes sense to combine both steps into a single operation. In the sequel, we extend the ICT to two-dimensional arrays and review the details of the combined normalization and quantization operations.

First we note that

$$C^{-1} = C^T\Delta \tag{2}$$

To verify this equation, we write $CC^{-1} = I$. Substitute $C = \sqrt{\Delta^{-1}}M$ to obtain $\sqrt{\Delta^{-1}}MC^{-1} = I$. Premultiplying this equation by $M^T\sqrt{\Delta}$ gives the desired result.

To recover the data vector $\mathbf{X}$ from the transform $\mathbf{Y}$, we premultiply $\mathbf{Y}$ by C^{-1}. Thus, we have

$$\mathbf{X} = C^{-1}\mathbf{Y} = C^{-1}C\mathbf{X} = \left(C^T\sqrt{\Delta}\right)\left(\sqrt{\Delta}C\right)\mathbf{X} \tag{3}$$

[1] In integer arithmetic calculations, the alternative expression $\lfloor (a + \lfloor q/2 \rfloor)/q \rfloor$ is often used.

This equation highlights $\sqrt{\Delta}C$ and $C^T\sqrt{\Delta}$ as matrix representations of the direct normalized ICT and its inverse transform, respectively.

The extension to two-dimensional blocks is straightforward. The two-dimensional ICT is a separable transform that can be computed by successively applying one-dimensional ICTs to the columns of the data block and then to the rows of the intermediate result. In matrix form, this is represented by premultiplying by the transform matrix and postmultiplying by its transpose. Letting X and Y denote the two-dimensional data block and its transform, we have

$$Y = \sqrt{\Delta}\, CXC^T\sqrt{\Delta} \tag{4}$$

and

$$X = C^T\sqrt{\Delta}\, Y\sqrt{\Delta}C \tag{5}$$

The above equations can be simplified by noting that CXC^T and Y are premultiplied and postmultiplied by the diagonal matrix $\sqrt{\Delta}$. Given two diagonal matrices, D_1 and D_2, and an arbitrary matrix, A, the product D_1AD_2 can be more efficiently calculated by term-by-term multiplying all the elements of A by the corresponding elements of $D_1\mathbf{1}D_2$, where $\mathbf{1}$ is a matrix of the same size as A, composed only of 1's. This equivalent procedure reduces the number of multiplications by one-half. Denoting the term-by-term multiplication by the operator symbol #, we write

$$Y = (CXC^T)\ \#\ N \tag{6}$$

and

$$X = C^T(Y\ \#\ N)\ C \tag{7}$$

where the normalization matrix is defined as

$$N = \sqrt{\Delta}\ \mathbf{1}\ \sqrt{\Delta} \tag{8}$$

It is clear in these equations that the original data block X can be exactly reconstructed from the transform block Y. In practice, though, the transform coefficients are quantized and reproduced with finite precision at the user end. As we have mentioned, the quantization procedure can be implemented by rounded term-by-term integer division of the transform coefficients by an array of quantization factors. This operation can be represented by the term-by-term multiplication of the transform matrix Y by H, the array of inverses of the quantization factors, it being understood that the results are rounded to the nearest integer.

Denoting the quantized transform by Y^*, we have

$$Y^* = Y\ (\#)\ H = (CXC^T)\ \#\ N\ (\#)\ H \tag{9}$$

where we use (#) to represent the nearest integer operation.

As it is easily verifiable, the above # operations can be permuted, and we can combine the normalization and quantization arrays into a single array $Q \triangleq N \# H$ so that

$$Y^* = (CXC^T)\ (\#)\ Q \tag{10}$$

The reconstructed image block X^* can be obtained by

$$X^* = C^T(Y^* \#\ Q^*)\ C \tag{11}$$

where Q^*, the counterpart of the array Q, satisfies the equation

$$Q^*\ \#\ Q = \Delta\ \mathbf{1}\ \Delta \tag{12}$$

We are now ready to consider the simplified ICT implementation. Suppose the optimized normalization/quantization matrix Q for a particular class of images has been obtained. Typically, the entries of this matrix are approximated by integer inverses. Let Q_a denote an alternative approximation of Q, with entries composed exclusively of negative powers of two. For the approximation rule, we select the nearest power of two, breaking ties in favor of the smaller factors. Substituting Q_a for Q, the quantization procedure can be implemented with simple binary shifts in the representations of the transformed coefficients, followed by addition of 0.5, and truncation.[2]

Performing the quantization by binary shifts may cut the execution time by more than 50 percent as compared with the time required by the operation with integer divisions. The exact time savings depends on the numbers involved and on the processor architecture. In general, binary shifts are executed in a fraction of the time required for integer divisions, which are usually executed by repeated shifts and subtractions.

To compensate for the approximation in Q, the reconstruction is performed with the matrix Q_a^* obtained from

$$Q_a^*\ \#\ Q_a = \Delta\ \mathbf{1}\ \Delta \tag{13}$$

III. Examples

To illustrate, we examine two examples with commonly used quantization templates.

A. Uniform Quantization

In the first example, we consider a uniform quantization template. This is one of the quantization templates to be used in the Galileo image-compression system. In this case, $H = \mathbf{1}$, so that Q and Q^* are equal to $N = \sqrt{\Delta}\ \mathbf{1}\ \sqrt{\Delta}$. In practice, Q is only an integer approximation of N, so that Q^* and Q are slightly different.

Recalling the expression for C, we have

[2] To avoid the addition of a noninteger, the alternative quantization rule given in Footnote 1 can be used.

$$D = CC^T = \begin{pmatrix} 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 78 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 40 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 78 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 78 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 40 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 78 \end{pmatrix} \tag{14}$$

$$D\,\mathbf{1}\,D = \begin{pmatrix} 64 & 624 & 320 & 624 & 64 & 624 & 320 & 624 \\ 624 & 6084 & 3120 & 6084 & 624 & 6084 & 3120 & 6084 \\ 320 & 3120 & 1600 & 3120 & 320 & 3120 & 1600 & 3120 \\ 624 & 6084 & 3120 & 6084 & 624 & 6084 & 3120 & 6084 \\ 64 & 624 & 320 & 624 & 64 & 624 & 320 & 624 \\ 624 & 6084 & 3120 & 6084 & 624 & 6084 & 3120 & 6084 \\ 320 & 3120 & 1600 & 3120 & 320 & 3120 & 1600 & 3120 \\ 624 & 6084 & 3120 & 6084 & 624 & 6084 & 3120 & 6084 \end{pmatrix} \tag{15}$$

and

$$\sqrt{D}\,\mathbf{1}\,\sqrt{D} \cong \begin{pmatrix} 8 & 25 & 18 & 25 & 8 & 25 & 18 & 25 \\ 25 & 78 & 56 & 78 & 25 & 78 & 56 & 78 \\ 18 & 56 & 40 & 56 & 18 & 56 & 40 & 56 \\ 25 & 78 & 56 & 78 & 25 & 78 & 56 & 78 \\ 8 & 25 & 18 & 25 & 8 & 25 & 18 & 25 \\ 25 & 78 & 56 & 78 & 25 & 78 & 56 & 78 \\ 18 & 56 & 40 & 56 & 18 & 56 & 40 & 56 \\ 25 & 78 & 56 & 78 & 25 & 78 & 56 & 78 \end{pmatrix} \tag{16}$$

Therefore, the normalization/quantization template is given by the matrix

$$Q = \begin{pmatrix} 8^{-1} & 25^{-1} & 18^{-1} & 25^{-1} & 8^{-1} & 25^{-1} & 18^{-1} & 25^{-1} \\ 25^{-1} & 78^{-1} & 56^{-1} & 78^{-1} & 25^{-1} & 78^{-1} & 56^{-1} & 78^{-1} \\ 18^{-1} & 56^{-1} & 40^{-1} & 56^{-1} & 18^{-1} & 56^{-1} & 40^{-1} & 56^{-1} \\ 25^{-1} & 78^{-1} & 56^{-1} & 78^{-1} & 25^{-1} & 78^{-1} & 56^{-1} & 78^{-1} \\ 8^{-1} & 25^{-1} & 18^{-1} & 25^{-1} & 8^{-1} & 25^{-1} & 18^{-1} & 25^{-1} \\ 25^{-1} & 78^{-1} & 56^{-1} & 78^{-1} & 25^{-1} & 78^{-1} & 56^{-1} & 78^{-1} \\ 18^{-1} & 56^{-1} & 40^{-1} & 56^{-1} & 18^{-1} & 56^{-1} & 40^{-1} & 56^{-1} \\ 25^{-1} & 78^{-1} & 56^{-1} & 78^{-1} & 25^{-1} & 78^{-1} & 56^{-1} & 78^{-1} \end{pmatrix} \tag{17}$$

and the reconstruction template Q^* is expressed as

$$Q^* = \begin{pmatrix} 1/8 & 25/624 & 18/320 & 25/624 & 1/8 & 25/624 & 18/320 & 25/624 \\ 25/624 & 1/78 & 56/3120 & 1/78 & 25/624 & 1/78 & 56/3120 & 1/78 \\ 18/320 & 56/3120 & 1/40 & 56/3120 & 18/320 & 56/3120 & 1/40 & 56/3120 \\ 25/624 & 1/78 & 56/3120 & 1/78 & 25/624 & 1/78 & 56/3120 & 1/78 \\ 1/8 & 25/624 & 18/320 & 25/624 & 1/8 & 25/624 & 18/320 & 25/624 \\ 25/624 & 1/78 & 56/3120 & 1/78 & 25/624 & 1/78 & 56/3120 & 1/78 \\ 18/320 & 56/3120 & 1/40 & 56/3120 & 18/320 & 56/3120 & 1/40 & 56/3120 \\ 25/624 & 1/78 & 56/3120 & 1/78 & 25/624 & 1/78 & 56/3120 & 1/78 \end{pmatrix}$$

Our proposed simplified ICT algorithm uses a power-of-two approximation for the elements of Q, so that Q is replaced by Q_a, given by

$$Q_a = \begin{pmatrix} 8^{-1} & 32^{-1} & 16^{-1} & 32^{-1} & 8^{-1} & 32^{-1} & 16^{-1} & 32^{-1} \\ 32^{-1} & 64^{-1} & 64^{-1} & 64^{-1} & 32^{-1} & 64^{-1} & 64^{-1} & 64^{-1} \\ 16^{-1} & 64^{-1} & 32^{-1} & 64^{-1} & 16^{-1} & 64^{-1} & 32^{-1} & 64^{-1} \\ 32^{-1} & 64^{-1} & 64^{-1} & 64^{-1} & 32^{-1} & 64^{-1} & 64^{-1} & 64^{-1} \\ 8^{-1} & 32^{-1} & 16^{-1} & 32^{-1} & 8^{-1} & 32^{-1} & 16^{-1} & 32^{-1} \\ 32^{-1} & 64^{-1} & 64^{-1} & 64^{-1} & 32^{-1} & 64^{-1} & 64^{-1} & 64^{-1} \\ 16^{-1} & 64^{-1} & 32^{-1} & 64^{-1} & 16^{-1} & 64^{-1} & 32^{-1} & 64^{-1} \\ 32^{-1} & 64^{-1} & 64^{-1} & 64^{-1} & 32^{-1} & 64^{-1} & 64^{-1} & 64^{-1} \end{pmatrix} \tag{18}$$

and the reconstruction template Q_a^* is expressed by

$$Q_a^* = \begin{pmatrix} 1/8 & 32/624 & 16/320 & 32/624 & 1/8 & 32/624 & 16/320 & 32/624 \\ 32/624 & 64/6084 & 64/3120 & 64/6084 & 32/624 & 64/6084 & 64/3120 & 64/6084 \\ 16/320 & 64/3120 & 32/1600 & 64/3120 & 16/320 & 64/3120 & 32/1600 & 64/3120 \\ 32/624 & 64/6084 & 64/3120 & 64/6084 & 32/624 & 64/6084 & 64/3120 & 64/6084 \\ 1/8 & 32/624 & 16/320 & 32/624 & 1/8 & 32/624 & 16/320 & 32/624 \\ 32/624 & 64/6084 & 64/3120 & 64/6084 & 32/624 & 64/6084 & 64/3120 & 64/6084 \\ 16/320 & 64/3120 & 32/1600 & 64/3120 & 16/320 & 64/3120 & 32/1600 & 64/3120 \\ 32/624 & 64/6084 & 64/3120 & 64/6084 & 32/624 & 64/6084 & 64/3120 & 64/6084 \end{pmatrix}$$

B. JPEG Quantization

The Joint Photographic Experts Group (JPEG) has put forth an image-compression standard based on the discrete cosine transform. A typical quantization array for this standard is given by [4]

$$J = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 60 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix} \tag{19}$$

Combining this array with the ICT normalization gives the matrix

$$Q = \begin{pmatrix} 128^{-1} & 275^{-1} & 179^{-1} & 400^{-1} & 192^{-1} & 999^{-1} & 912^{-1} & 1524^{-1} \\ 300^{-1} & 936^{-1} & 782^{-1} & 1482^{-1} & 649^{-1} & 4524^{-1} & 3351^{-1} & 4290^{-1} \\ 250^{-1} & 726^{-1} & 640^{-1} & 1341^{-1} & 716^{-1} & 3184^{-1} & 2760^{-1} & 3128^{-1} \\ 350^{-1} & 1326^{-1} & 1229^{-1} & 2262^{-1} & 1274^{-1} & 6786^{-1} & 3351^{-1} & 4836^{-1} \\ 144^{-1} & 550^{-1} & 662^{-1} & 1399^{-1} & 544^{-1} & 2723^{-1} & 1843^{-1} & 1923^{-1} \\ 600^{-1} & 2730^{-1} & 3072^{-1} & 4992^{-1} & 2023^{-1} & 8112^{-1} & 6312^{-1} & 7176^{-1} \\ 877^{-1} & 3575^{-1} & 3120^{-1} & 4860^{-1} & 1843^{-1} & 6759^{-1} & 4800^{-1} & 5642^{-1} \\ 1799^{-1} & 7176^{-1} & 5306^{-1} & 7644^{-1} & 2798^{-1} & 7800^{-1} & 5753^{-1} & 7722^{-1} \end{pmatrix}$$

and its counterpart

$$Q^* = \left(\begin{matrix} 128/64 & 275/624 & 179/320 & 400/624 & \ldots \\ 300/624 & 936/6084 & 782/3120 & 1482/6084 & \ldots \\ 250/320 & 726/3120 & 640/1600 & 1341/3120 & \ldots \\ 350/624 & 1326/6084 & 1229/3120 & 2262/6084 & \ldots \\ 144/64 & 550/624 & 662/320 & 1399/624 & \ldots \\ 600/624 & 2730/6084 & 3072/3120 & 4992/6084 & \ldots \\ 877/320 & 3575/3120 & 3120/1600 & 4860/3120 & \ldots \\ 1799/624 & 7176/6084 & 5306/3120 & 7644/6084 & \ldots \end{matrix}\right.$$

$$\left.\begin{matrix} \ldots & 192/64 & 999/624 & 912/320 & 1524/624 \\ \ldots & 649/624 & 4524/6084 & 3351/3120 & 4290/6084 \\ \ldots & 716/320 & 3184/3120 & 2760/1600 & 3128/3120 \\ \ldots & 1274/624 & 6786/6084 & 3351/3120 & 4836/6084 \\ \ldots & 544/64 & 2723/624 & 1843/320 & 1923/624 \\ \ldots & 2023/624 & 8112/6084 & 6312/3120 & 7176/6084 \\ \ldots & 1843/320 & 6759/3120 & 4800/1600 & 5642/3120 \\ \ldots & 2798/624 & 7800/6084 & 5753/3120 & 7722/6084 \end{matrix}\right) \tag{20}$$

Therefore, a possible power-of-two approximation for Q is given by

$$Q_a = \begin{pmatrix} 128^{-1} & 256^{-1} & 128^{-1} & 512^{-1} & 128^{-1} & 1024^{-1} & 1024^{-1} & 1024^{-1} \\ 256^{-1} & 1024^{-1} & 512^{-1} & 1024^{-1} & 512^{-1} & 4096^{-1} & 4096^{-1} & 4096^{-1} \\ 256^{-1} & 512^{-1} & 512^{-1} & 1024^{-1} & 512^{-1} & 2048^{-1} & 2048^{-1} & 2048^{-1} \\ 256^{-1} & 1024^{-1} & 1024^{-1} & 2048^{-1} & 1024^{-1} & 4096^{-1} & 4096^{-1} & 4096^{-1} \\ 128^{-1} & 512^{-1} & 512^{-1} & 1024^{-1} & 512^{-1} & 2048^{-1} & 2048^{-1} & 2048^{-1} \\ 512^{-1} & 2048^{-1} & 2048^{-1} & 4096^{-1} & 2048^{-1} & 8192^{-1} & 4096^{-1} & 8192^{-1} \\ 1024^{-1} & 4096^{-1} & 2048^{-1} & 4096^{-1} & 2048^{-1} & 8192^{-1} & 4096^{-1} & 4096^{-1} \\ 2048^{-1} & 8192^{-1} & 4096^{-1} & 8192^{-1} & 2048^{-1} & 8192^{-1} & 4096^{-1} & 8192^{-1} \end{pmatrix}$$

and the corresponding reconstruction array is

$$Q_a^* = \left(\begin{matrix} 128/64 & 256/624 & 128/320 & 512/624 & \ldots \\ 256/624 & 1024/6084 & 512/3120 & 1024/6084 & \ldots \\ 256/320 & 512/3120 & 512/1600 & 1024/3120 & \ldots \\ 256/624 & 1024/6084 & 1024/3120 & 2048/6084 & \ldots \\ 128/64 & 512/624 & 512/320 & 1024/624 & \ldots \\ 512/624 & 2048/6084 & 2048/3120 & 4096/6084 & \ldots \\ 1024/320 & 4096/3120 & 2048/1600 & 4096/3120 & \ldots \\ 2048/624 & 8192/6084 & 4096/3120 & 8192/6084 & \ldots \end{matrix}\right.$$

$$\left.\begin{matrix} \ldots & 128/64 & 1024/624 & 1024/320 & 1024/624 \\ \ldots & 512/624 & 4096/6084 & 4096/3120 & 4096/6084 \\ \ldots & 512/320 & 2048/3120 & 2048/1600 & 2048/3120 \\ \ldots & 1024/624 & 4096/6084 & 4096/3120 & 4096/6084 \\ \ldots & 512/64 & 2048/624 & 2048/320 & 2048/624 \\ \ldots & 2048/624 & 8192/6084 & 4096/3120 & 8192/6084 \\ \ldots & 2048/320 & 8192/3120 & 4096/1600 & 4096/3120 \\ \ldots & 2048/624 & 8192/6084 & 4096/3120 & 8192/6084 \end{matrix}\right) \tag{21}$$

It is interesting to note that the power-of-two approximation can also be considered in a dual application, where the encoder is capable of performing floating point arithmetic at high speed, and the compressed images need to be recovered at a large number of low-cost receivers. In this situation, which could be encountered, for example, in broadcast video applications, it would be beneficial to replace the reconstruction array Q^* given in Eq. (20) by a power-of-two approximation. This approximation would have to be precompensated by a quantization array composed of real numbers, such that the term-by-term product of the two arrays produces Δ **1** Δ [cf. Eq. (13)].

IV. Practical Results

The simplified algorithm was compared with the conventional ICT algorithm in the compression of three 800 × 800, 8-bit typical planetary images. The comparisons were made in terms of the rate-distortion trade-off, expressed in plots of the peak signal-to-noise ratio (PSNR) as a function of the achieved compression ratio (CR). The PSNR of 8-bit images is defined by $10 \log_{10}(255^2/MSE)$, where MSE denotes the mean square error of the reconstructed image. The CR is defined as the ratio between the number of bits in the original image and the number of bits in the compressed image.

Figure 1 shows the rate-distortion performance obtained for image "ant9," with the standard ICT and the simplified algorithms. The original is a noisy, hard-to-compress image, with a large number of streaks caused by α-particle effects on the CCD sensor array.

The quantization array (H) is uniform, and the set of 7 points corresponding to each algorithm was obtained by weighting the combined normalization/quantization array given in Eqs. (16) and (17) by 1, 2, 4, 8, 16, 32, and 64. The weights were chosen as powers of two, since the use of arbitrary weights would obviously interfere with the practicality of the simplified algorithm. A smoother variation of the rate-distortion points can be obtained by progressively weighting pieces of the quantization array. For example, the bottom-right half of the array, which corresponds to higher frequency terms, could be weighted first, before the adjustment is extended to the rest of the array. Clearly, a large number of weighting strategies are possible and their relative performances will depend on the image being compressed.

From Fig. 1, we notice that the two algorithms are quite comparable in performance. The maximum separation between the interpolated curves, occurring at the higher compression ratios, is approximately one-third of a dB in PSNR (approximately a 7-percent variation in MSE). Figure 2 presents a similar plot for image "r4," a somewhat "busy" image with little sensor noise. Figure 3 shows results with "saturn1," an easier image to compress, reaching a compression ratio of 60 at the reasonable PSNR of 34 dB. In both cases, the observed performances obtained with uniform quantization are very similar, the maximum separation between the interpolated curves amounting to about one-quarter of a dB, or a 6-percent variation in MSE.

To exemplify the potential savings in computation time, we consider the processor of Galileo's attitude and articulation control subsystem (AACS) [5], to be used also for onboard image compression. The times required for an addition and an integer division are 5 and 13.25 μsec, respectively. Thus, normalization and quantization of a conventional 8 × 8 ICT block requires 64 × 18.25 μsec = 1.168 msec (one addition and one division per transform coefficient, according to the formula in Footnote 1). Alternatively, a fixed-length binary shift operation takes from 2.5 to 6.5 μsec, depending on the length. Estimating an average duration of 4.5 μsec for binary shifts, the time required for normalization/quantization of an 8 × 8 ICT block is 64 × 9.5 μsec = 608 μsec with the simplified implementation. Thus, a savings of 560 μsec per ICT block can be realized. Nevertheless, two factors contribute to making the proposed scheme not applicable to Galileo. First, the AACS processor does not take full advantage of the binary shift implementation. There are no instructions for variable-length binary shifts, so additional time is required for setting the length. Second, a significant fraction (74 percent) of the average time needed for compressing images with the AACS processor is spent on the entropy encoding stage (a combination of run length and Huffman code), due to the intense memory accessing required.

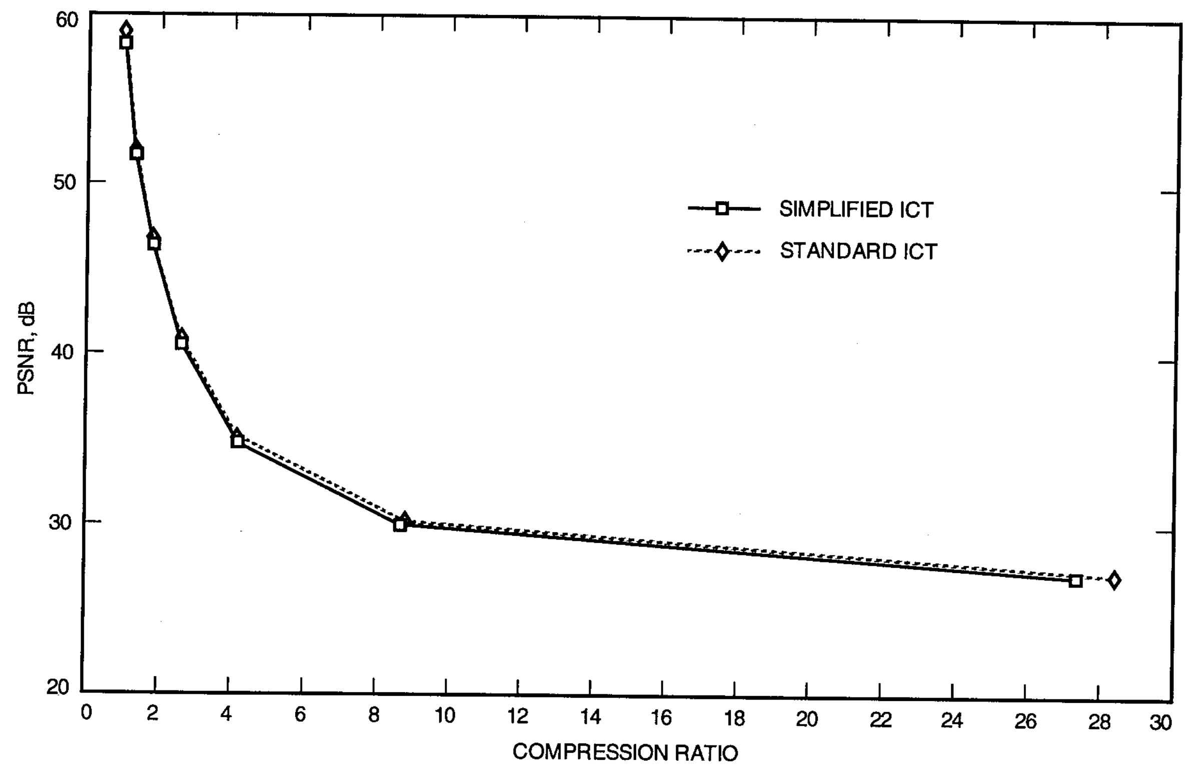

Fig. 1. PSNR as a function of compression ratio for image "ant9," with uniform quantization.

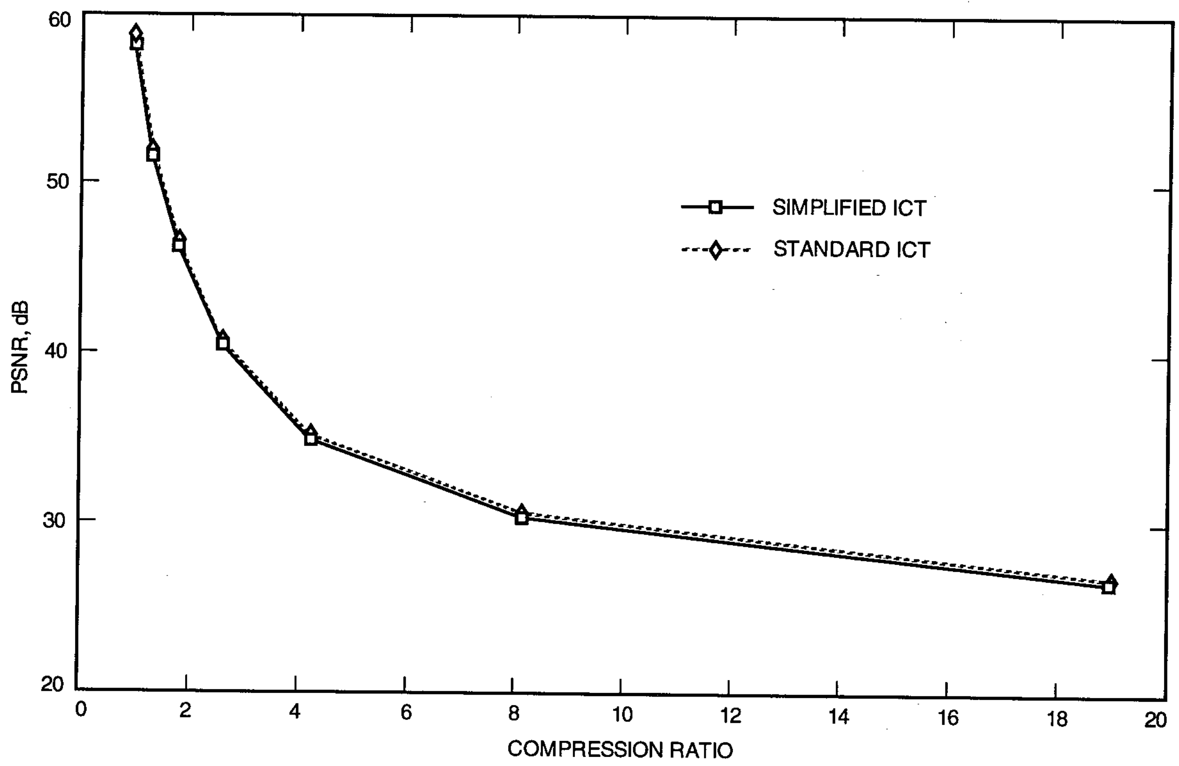

Fig. 2. PSNR as a function of compression ratio for image "r4," with uniform quantization.

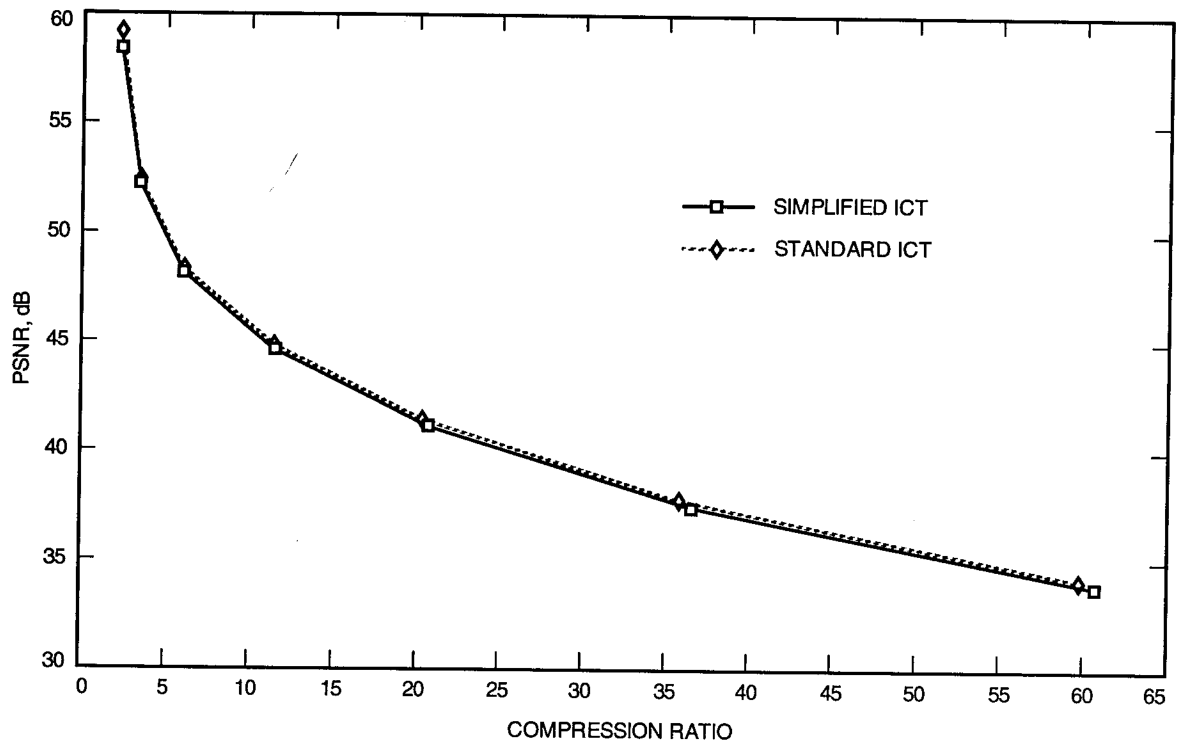

Fig. 3. PSNR as a function of compression ratio for image "saturn1," with uniform quantization.

V. Conclusions

This article described a simplification of the integer cosine transform. The simplified transform is orthogonal and approximately normalized. This transform is considered in the context of an image compression system in which normalization and quantization operations are performed jointly, exclusively by means of additions and binary shifts. The departure from a perfectly normalized transform is compensated for in the inverse transform operation, performed with real arithmetic. Alternatively, the power-of-two approximation can be used at the inverse transform operation, having been preconditioned by the proper real arithmetic during the direct transform calculation. The algorithm is useful in applications with disparate speed and complexity constraints at the points where the direct and inverse transforms are calculated. The performance of the simplified transform is compared with that of a conventional ICT in the compressing of a set of three planetary images. The distortion-rate characteristics of the two algorithms are found to be very similar, differing only by a fraction of a dB in PSNR. The advantage of using the simplified transform is the reduction in time required for normalization and quantization. This reduction is generally processor-dependent and can be significant in special-purpose hardware implementations.

References

[1] W.-K. Cham, "Development of Integer Cosine Transforms by the Principle of Dyadic Symmetry," *IEE Proceedings*, vol. 136, pt. I, no. 4, pp. 276–282, August 1989.

[2] K.-M. Cheung, F. Pollara, and M. Shahshahani, "Integer Cosine Transform for Image Compression," *The Telecommunications and Data Acquisition Progress Report 42-105, vol. January-March 1991*, Jet Propulsion Laboratory, Pasadena, California, pp. 45–53, May 15, 1991.

[3] K.-M. Cheung and K. Tong, "Proposed Data Compression Schemes for the Galileo S-Band Contingency Mission," *1993 Space and Earth Science Data Compression Workshop*, NASA Conference Publication 3191, Snowbird, Utah, pp. 99–109, April 2, 1993.

[4] W. Pennebaker and J. Mitchell, *JPEG Still Image Compression Standard*, New York: Van Nostrand Reinhold, 1993.

[5] *HAL/S ATAC 16M Reference Documents*, Document 52-054614, Applied Technologies Advanced Computer.